针对不同硬件平台使用不同优化机制使程序更加适应其体系结构，同时实现松耦合、高内聚的模块集成的具有完全自主知识产权的大型流体机械并行计算软件

何锋的重要贡献：

**为充分利用E级计算系统节点间、节点内处理核之间计算资源，提出基于互斥锁的MPI/OpenMP粗粒度混合并行，实现了节点间消息通信、节点内共享内存的高效通信模式，完成了大规模多级混合并行性能优化。**

1）在MPI层采用持久化通信方式；在 CFD 数值仿真过程中 MPI 通信的调用过程是在循环迭代中反复进行的，大规模并行时，这种反复创建和销毁 MPI 通信通道的开销也不容忽视。根据流体机械的物理模型特点，许多静态网格区域在分解以后， 通信拓扑中的通信对象就不会在发生任何变化。初始化时创建持久化通信通道，提供持久不变的通信发送缓冲区和接收缓冲区，以及通信的对象。

2）基于互斥锁、双缓冲和自旋式非阻塞机制设计细粒度线程间同步方法，优化了 OpenMP 共享内存通信过程；由于 OpenMP 本身提供的同步方法粒度较粗，全局同步带来的开销

成为共享内存通信的瓶颈。如果利用简单的显式全局实现线程间同步，由于各个核的计算速度受实际环境的影响运行速度差异较大，而且随着进程内线程数量的增加，显式全局同步带来的时间开销可能会超过线程本身共享内存的通信时间。每个边界面的通信过程本身只涉及两个线程，每个边界面通信需要同步两个线程，设置保证“写后读”的写完锁和保证“读后写”的读完锁。通信过程是逐个边界面串行推进，自旋式非阻塞同步机制 非阻塞的方式轮询所有边界面。 成功加锁就进行通信，不成功则先跳过。不会因为某个边界面加锁不成功而阻塞整个线程。 随着自旋查询的进行， 未通信的边界面也会来越少，最终完成所有边界面的共享内存通信过程。

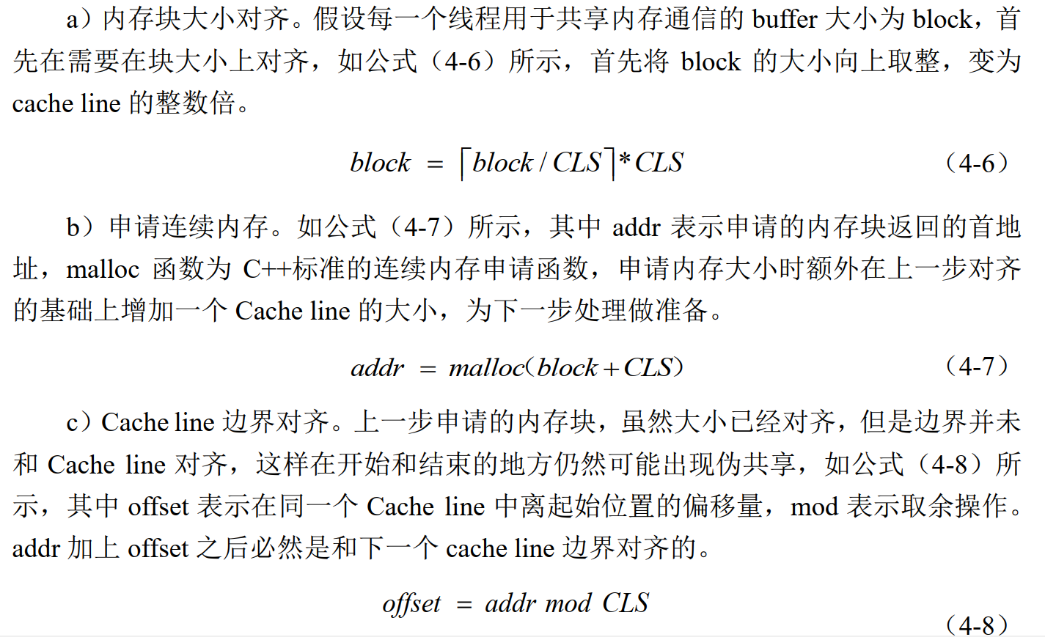
3）采用数据 padding、内存边界对齐和读/写、写/写分离的方法避免 cache 一致性导致的 Cache line 竞争，改善了 Cache line 伪共享现象

假设在两个处于同一个多核处理器上一个在计算核 C0 运行上的线程想要更新变量 A 的值，同时另外一个在计算核 C1 运行的线程想要更新变量 B 的值。假设这两个变量都位于同一个缓存行 Cache line， 此时两个线程就会同时发送 RFO（Request For occupy）消息， 想要获得该缓存行的占有权。表面上 A 和 B 都是被独立线程操作的变量，而且两操作之间也没有任何依赖关系，似乎是在并行进行。 但是由于处于同一个 Cache line，则在实际的运行中变为串行，这种变化在编码时很难以发现，且和 CPU 的架构如 Cache line 大小、 Cache Level 等紧密相关，是并行程序性能的隐形杀手。

物理内存按照 Cache 映射规则，被划分为一个个 Cache line 大小的内存块。 根据Cache line 的大小 CLS 和物理内存块的首地址和大小，就可以计算出该块物理内存所在的 Cache line。 从内存分配的角度出发，在分配阶段就完成 Cache line 的对齐，就可以保证每一个线程共享内存通信的 buffer 为 Cache line 独占， 从而避免与其他线程载入同一个 Cache line。

Cache line 只有在被改写之后才会发生状态变化，引起一系列的一致性操作。 如果两个线程都是读操作则不会产生伪共享，因此从读写的角度出发，将两个线程的写/写或者读/写操作的内存块相分离，设置一定的距离也不会载入同一个 cache line 进行更新操作，也能改善 cache 的伪共享。

NUMA 体系架构下，虽然片上的 CPU 通过 QPI 总线可以实现互相访问，但是由于物理链路的远近，会产生远程命中。首先是远程 cache 命中，除私有的 Cache 的层级结构以外，还存在远程 Cache 命中访问的情况。计算核发起 Cache 访问，在 LLCCache 仍然 Miss，但却被告知该块内存已经载入另一个 core 的私有 Cache 内， 这就远程命中。其次是远程访存，由于一个节点内的所有 CPU 理论上统一编址，可以访问，但是如果访存命中另一个片上的内存，由于距离较远，就会产生成倍的延迟。

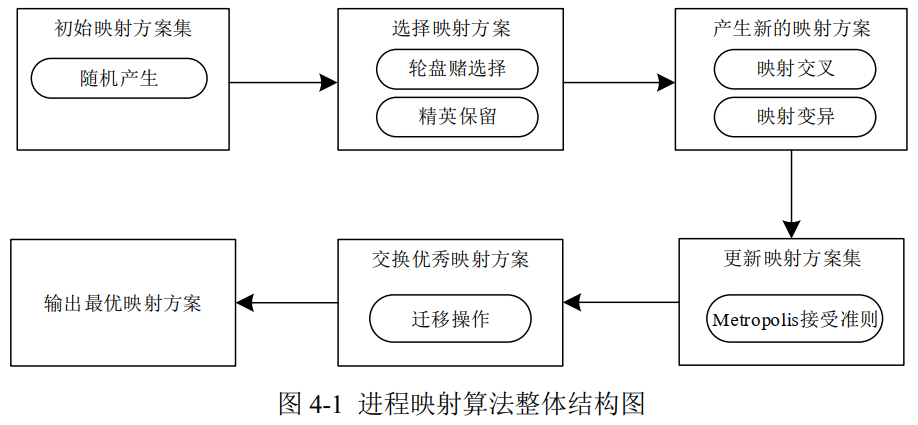


安伟华贡献：

1）通过对 CFD并行计算软件进行源码剖析和TAU（Tuning and Analysis Utilities）性能测试软件来动态追踪来构建通信量与通信次数的通信模式矩阵，通过测试计算资源间的

2）实际通信性能来构建通信距离矩阵。采用 ping-pong 测试来获取不同计算资源间的实际通信指标，包括通信延迟和通信带宽，短消息时候通信延迟作为通信距离，长消息使用带宽的导数作为通信距离

3）基于并行遗传算法的搜索

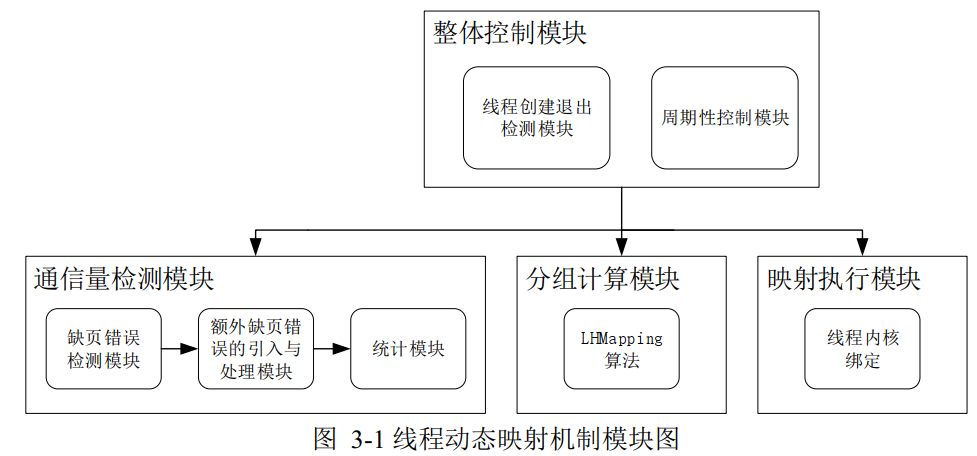


首先，由改进的梅森旋转随机数生成算法生成多个子映射方案集， 每个映射方案集中都包含多个进

程映射序列，每一个序列对应一种进程映射方案；

然后产生变异，并根据模拟退火算法中的 Metropolis 接受准则来对新产生的进程映射序列做取舍，更新映射方案集。加入模拟退火的思想能够保留一些通信开销较大的进程映射方案，能够保证映射方案集的多样性，避免映射算法陷入局部最优解；最后， 各个子映射方案集通过迁移操作来交换优秀的映射方案， 在其中筛选出当前最优的一组进程映射方案输出。

鲁晨欣贡献：



1）基于缺页错误的线程间通信量检测、

多线程并行应用程序运行时，某线程访问某存储页面的数据时产生了缺页错误， 反映了该线程需要访问该页面的数据，而该页面却不在内存中，或者该页面由内存管理单元控制的页表项失效了。利用这种特征， 检测机制只要能检测到两个不同线程先后在访问相同页面时产生了缺页错误，则表示他们需要访问相同页面的数据，于是检测机制认为这两个线程之间产生了一次通信事件。因此通过检测程序运行时操作系统中产生的缺页中断， 记录每次缺页中断时的线程 ID 和内存物理地址， 映射机制可以间接检测两个线程之间产生的通信事件。

2）线程映射的分组计算 映射分组算法 LHMapping（Loop-based Hierarchical Mapping Algorithm）

映射分组计算的目的是把通信量相对较大的线程分到同一个组中，实现线程与硬件对象的一一对应。这样在后续映射执行时，会把同一组内的线程映射到共享同一个硬件存储单元的 CPU 核心上。

2.1 Hwloc 工具测出体系结构硬件对象信息，多核共享的 Socket， 共享 Memory， 共享 Cache层次化分组方式

2.2 在进行分组时，算法以自底向上的方式，先将需要分布到每个 CPU 核心的线程，根据共享 L2 Cache 或共享 L3 Cache 可以共享的核数 K，每 K 个线程分为一组；底层分组完毕，接着处理到再上一层共享硬件对象的分组，即把最低层分好的线程组再分为容纳若干小线程组的大线程组；依此类推，直到分组到达根节点。组内基于通信量最大的贪心进行组内选择，

3）线程绑定：利用CPU 亲和性进行线程绑定。CPU 亲和性是指使进程或线程在某个给定的 CPU 上尽量长时间的运行而不被迁移到其他处理器的倾向性。

60万核：

每次的时间就是平均一步虚拟时间时间步的时间，

用户界面：

基于Qt用户界面，跨平台，

利用paramiko模拟用户远程登录超级计算机，**设计了面向神威太湖之光、天河3原型机、天河2A、天河1超算平台与普通HPC的多平台的作业管理接口**

后处理软件：paraview visualizer 部署paraview的web服务，进行后处理展示。